FIG. 2 illustrates, according to one embodiment, a flow chart of a user seeking to share candidate content with a recipient for the content, e.g., another user.

As illustrated, the user starts **200** a sharing application program. It will be appreciated that many different network application programs may be used to share data, including Napster, GNapster, Gnutella (a Napster-like application for sharing audiovisual content), and Internet browsers, such as Internet Explorer by Microsoft Corporation, Netscape Navigator by America Online, Inc., or another network communication application program. It is assumed the sharing application program has an appropriate plug-in or built-in capability for performing the invention.

The user selects **202** a candidate content for sharing, and a portion of the candidate content is selected **204** from which feature data is extracted **206** as discussed above in FIG. 1. For the purposes of this discussion, it is assumed the selected candidate content is a video encoding, and that selecting **204** comprises selecting regions of multiple video frames from which to extract **206** feature data as discussed above for FIG 1. However, it will be appreciated that the invention may be practiced with other data formats. Once the feature data is extracted, the next step is to attempt to identify the candidate content which the user is attempting to share.

It will be appreciated that, depending on the nature of the candidate content, feature extraction may require decoding the candidate content. For example, for an MPEG video encoding, in order to extract feature data from the first frame of the selected portion, it may be necessary to decode the MPEG stream up to the first frame. Such decoding requirements, when necessary, are implied in the present description and will not be called out explicitly.

Assume a server or other machine is communicatively coupled to a data store storing extracted feature data for different content. In one embodiment, the client sends **208** the server, e.g., by way of a plug-in, built-in or other communication means, the feature data extracted from the candidate content. The server retrieves **210** stored (see FIG. 1 item **108**) feature data for a reference content, and compares **212** the extracted **206** feature data with the stored **108** feature data to see if there is a match. In one embodiment, to foil attempts to circumvent sharing protection, the server performs a sliding window comparison against the extracted feature data for the candidate content (see FIG. 3).

A test **214** is performed to determine whether the candidate content matched the current reference content. If so, then in one embodiment, sharing is denied **216**, and the sharing application program instructs the user of the program regarding any options, if any. For example, as discussed in FIG. 1, there may be associated data for a reference content that provides options to the user of the sharing application program or to the intended recipient of the candidate content, e.g., purchase information may be provided to the intended recipient and/or the user. It will be appreciated that any arbitrary action may be taken in response to determining a match.

If the candidate content did not match the reference content, then a test **218** is performed to determine whether there are other known reference content having feature data stored in the data store. If so, then another reference content is selected **220** for comparing against the candidate content, and processing continues with retrieving **210** the stored feature data for the newly selected reference content. If there were no more reference content feature data in the data store, then the candidate content is not

known to be protected, and the sharing is allowed **222** to proceed. For example, the user's sharing application program can be instructed by the server to proceed with a data transfer of the candidate content.

In one embodiment, not illustrated, extracted **206** feature data for the candidate content is saved to allow later identification and/or validation of a previous transfer based on new values entered into the data store.

FIG. 3 illustrates one embodiment for implementing the FIG. 2 comparison **212** between extracted feature data for a candidate content and a reference content. In the illustrated embodiment, video frames from both the reference content and candidate content have been analyzed and feature data extracted. It is assumed that all of the reference content frames have been analyzed and extracted feature data stored in a data store, while only a subset of the candidate content frames have been analyzed.

A sliding-window comparison is performed to identify the appropriate starting frame, if any, within the reference content that corresponds to the first frame of candidate content that was analyzed and feature data extracted. Performing a sliding window comparison increases security, in that the starting point for analysis of the candidate content can be arbitrarily selected. In the illustrated embodiment, indexes (or pointers) are used to identify a particular frame of the reference content or candidate content for which feature data has been extracted. It will be appreciated that other techniques may be used to traverse and compare feature data.

In one embodiment, therefore, a reference index is set **300** to a first frame of the extracted feature data for the reference content, and a candidate index is set **302** to a